Distributed Placement of Autonomic Internet Services

Panagiotis Pantazopoulos, Merkouris Karaliopoulos, *Member, IEEE,*
and Ioannis Stavrakakis, *Fellow, IEEE*

Abstract—The optimal placement of service facilities largely determines the capability of a data network to efficiently support its users' service demands. As centralized solutions over large-scale distributed environments are extremely expensive, inefficient or even infeasible, distributed approaches that rely on partial topology and demand information are the only credible approaches to the service placement problem, even at the expense of non-guaranteed optimality. In this paper, we propose a distributed service migration heuristic that iteratively solves instances of the 1-median problem pushing progressively the service to more cost-effective locations. Key to our algorithm is a traffic-aware centrality metric, called weighted conditional betweenness centrality (wCBC), that captures the ability of a node to act as service demand concentrator and is employed in both selecting the nodes and setting their weights for the 1-median problem instance. The assessment of our heuristic proceeds in two steps. First, assuming (ideal) knowledge of the invoked wCBC metric, we carry out a proof-of-concept study that demonstrates the effectiveness of the heuristic over synthetic and real-world topologies as well as its advantages against comparable local-search-like migration schemes. Next, we devise *practical* protocol implementations that approximate the heuristic using *local* measurements of transit traffic and preserve the excellent accuracy and fast convergence properties of the algorithm for different routing policies. Our solution applies to a broad range of networking scenarios, and is very relevant to the emerging trends for in-network storage and involvement of the end-user in the creation and distribution of lightweight (autonomic) service facilities.

1 Introduction

One of the most significant changes in networked communications over the last few years concerns the role of the *end-user*. Traditionally the end-user has been almost exclusively the *consumer* of content and services generated by explicit entities referred to as content and service providers, respectively. Nowadays, Web2.0 technologies have enabled a paradigm shift towards more user-centric approaches to content generation and provision. This shift is strongly evidenced in the abundance of *User-Generated Content* (UGC) in social networking sites, blogs, wikis, or video distribution sites such as YouTube, which have motivated even the rethinking of the Internet architecture fundamentals [1], [2]. The generalization of the UGC concept towards services is increasingly viewed as one of the major trends in user-oriented networking [3].

The user-oriented service creation concept aims at engaging end-users in the generation and distribution of service components, more generally service facilities [4]. There already exist online applications that enable end-users to compose their own customized combination of heterogeneous web sources through easy-to-use graphical interfaces. Google App Engine [5] and Yahoo! Pipes [6] are typical examples of what is often referred to as *web-based mashup tools*. At the same time, efforts are under way to develop platforms that will engage end users in the creation of services with telecom-based features (*i.e.*, messaging, voice calls etc) integrated over the Next Generation Networks [7].

In parallel with the proliferation of the so-called *User-Generated Service* (UGS) paradigm, significant research effort is being carried out on the design and deployment of energy-efficient data storage architectures. Nano-datacenters have been proposed with the aim to offload part of the data management operations from the conventional power-hungry data-centers [8]. Numerous ISP-owned home gateways can be instrumented through virtualization technologies to host those lightweight peer servers and create a distributed Internet service platform that leverages end-user proximity. This shift towards more distributed data storage paradigms is further evidenced in a) the emerging Information-centric networking (ICN) paradigm [9] and its "in-network storage" argument; b) the realization of distributed-fashion social networks. In ICN, the network is equipped with functionality that allows it to contribute actively and reliably to the distribution of information objects. Likewise, Diaspora presents an instance of a social network implemented over interconnected nodes (*i.e.*, pods) that are hosted by numerous individual users on dedicated local storage. Each node operates an instance of the Diaspora software that turns it into a personal server [10].

In the intersection of the aforementioned trends emerges a rich ecosystem of highly autonomic service facilities that will be generated in pretty much every network location and considered independent of other software entities; many of these services will have strongly local scope, and will require, in principle, access to storage resources in various network locations. The technical challenge then is how to optimally place these services to minimize their access cost. However, more than ever before, the search is for *scalable* distributed service placement approaches that can be car-

• *The authors are with the Dept. of Informatics, University of Athens. E-mail: {ppantaz, mkaralio, ioannis}@di.uoa.gr*

This work has been partially supported by the EC IST-FET RECOGNITION project (FP7-IST-257756) and the EC Network of Excellence in Internet Science project EINS (FP7-ICT-288021).

ried out by typical network devices, without specialized processing capacity.

Motivated by these challenges, our work proposes a scalable decentralized heuristic algorithm that iteratively moves services from their generation location to the network location that minimizes their access cost. We follow earlier work on service placement in viewing the problem as an instance of the *facility location* problem [11]; precisely, we employ the 1-median formulation that is deemed more suitable for the user-centric service paradigm. Contrary to centralized approaches, where a single super-entity with global information about network topology and service demand solves the problem in a single iteration, we let it *migrate* towards its optimal location over a few hops. In each hop, a *small-scale* and simpler 1-median problem is solved so that the computational load is spread amongst the nodes along the migration path.

The service migration path is derived by invoking a node centrality metric we have devised earlier in [12] and call *weighted Conditional Betweenness Centrality (wCBC)*. This metric assesses the capacity of a network node to route service demand load from the rest of the network towards the current service location. Therefore, the metric can effectively identify directions (fig. 1.b) of high demand attraction, *i.e.*, network areas presenting high demand for the service. The metric help us competely determine the subgraph wherein the small-scale 1-median problem will be solved (*1-median subgraph*): first, by selecting the nodes of the subgraph and, then, by modulating the demand weights with which each one participates in the 1-median problem formulation. We detail the metric and the way it is used by our algorithm which we call cDSMA, in Sections 3 and 4.

Our contributions are both on the theoretical and practical front. From theoretical point of view, we propose a novel heuristic algorithm for the well-studied 1-median problem, which comes under the broader family of local-search techniques. The algorithm's convergence and approximation properties are discussed in Section 4. The negative result in this respect is that cDSMA is not a constant-ratio approximation algorithm, since synthetic examples can be constructed, where its deviation from the optimal cannot be bounded. On a practical note, we provide a systematic specification and evaluation of our algorithm, from the initial concept and properties down to practical implementation concerns as presented is Sections 6 and 7.

First, we carry out a proof-of-concept analysis (Section 6) over synthetic network topologies and under the ideal assumption that nodes can obtain accurate topological and demand information for the whole network. Essentially, this analysis tests the effectiveness of our metric as a guide of the service migration and exposes main properties and advantages of cDSMA. Next, maintaining these ideal conditions, the algorithm is shown to achieve remarkably high accuracy and fast convergence over real-world ISP topologies of hundreds of nodes, even when the 1-median problem iterations are solved with no more than 6% of the total network nodes. Hence, in realistic settings, and contrary to the theoretical worst-case prescriptions, cDSMA shows excellent potential to approximate the optimal solution. Moreover, it demonstrates remarkable scalability and robustness properties to service demand estimation inaccuracies across the network. Finally, it needs much fewer migration hops to yield placements of given accuracy than pure local-search policies, which seek for the next service migration hop within the local neighborhood of its current location (fig. 1.a).

Later in Section 7, we relax the assumption of ideal global information and propose a real-world distributed implementation for cDSMA, catering for all challenges related to distributed operation: how the node each time hosting the service collects topological and demand information and how it uses it to reconstruct the inputs needed by the algorithm. The implementation leverages the straightforward interpretation of the wCBC metric so that each node can locally obtain estimate values of its own wCBC and communicate them via dedicated messages to the current service host. This information can then be processed by the service host to extract *partial* topological information about the 1-median subgraph and determine the next service host on its migration path. The implementations can exercise further flexibility regarding how many nodes will measure and report their local estimates to the service host node. As shown in Section 8, this way they *effectively* tradeoff the algorithm accuracy with the generated message overhead.

2 The Service placement problem

The optimal placement of service facilities within network structures has been typically tackled as an instance of the facility location problem [11]. Input to the problem is the topology of the network nodes that may host services, their costs of installation and the distribution of service demand across the network users. The objective is to place services in a way that minimizes the joint cost of their installation and access over all users. Installation costs however are more relevant to settings of diverse capacity nodes where the service set-up cost can amount to a non-negligible part of a node's computational capacity. Such is the case of the traditional server-client model rather than the considered distributed environment of equally powerful nodes. Here, the lightweight services are expected to impose the same minimal set-up cost across all nodes. Hence, the optimal placement of up to k replicas of service instances is treated as a k-median problem [11]. Otherwise, when storage is restricted, a knapsack problem [13] formulation is employed.

We focus on the 1-median formulation that seeks to minimize the access cost of a single service replica since it matches better the expected features of the User-Generated Service paradigm. Recent evidence confirms the existence of few highly popular service/content objects and many others of interest to significantly fewer users [14]. UGS will enable the generation of service facilities in various network locations from a highly versatile set of amateur user-service providers. The huge majority of these lightweight service instances will be requiring minimum storage resources and addressing users in the "proximity" of the user-service provider, either geographical or social (friends, colleagues, *etc.*), so that their replication across

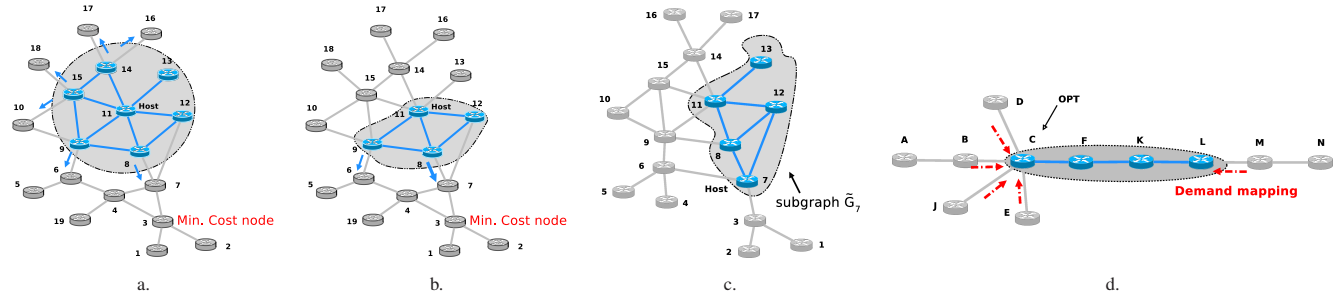


Fig. 1. a,b)1-median subgraph nodes under local-search heuristics(a) and cDSMA(b). c) With node 7 as current service host, two nodes (8 and 11) in the highlighted subgraph $\widetilde{G}_7$ map demand from the rest of the network (terms $w_{map}(8;7)$, $w_{map}(11;7)$). d) By crediting the demand of the $G \setminus \widetilde{G}_{Host}$ nodes only on the entry nodes C and L of the highlighted $\widetilde{G}_F$ subgraph, cDSMA moves the service to the optimal location C.

the network would not be justified[1].

We assume that the network topology is represented by an undirected connected graph $G = (V, E)$ of $|V|$ nodes and $|E|$ links. A subset $V_S \subseteq V$ of the total network nodes are enabled (or even willing) to act as service host sites and along with the set $E_S \subseteq E$ of edges linking them, form the, generally disconnected, subgraph $\widetilde{G} = (V_S, E_S)$. Each potential service host $k \in V_S$ may serve one or more users attached to some network node $n \in V$ and accessing the service with different intensity, generating demand $w(n)$ for it. The goal is to minimize over all network users the access cost of a service facility, which is

$$Cost(k) = \sum_{n \in \mathcal{V}} w(n) \cdot d(k, n) \qquad (1)$$

when the service is located at node $k \in V_S$. The distances $d(k, n)$ may have different context, depending on routing policies and the network dimensioning process. The exposition of the algorithm hereafter assumes that minimum cost paths coincide with minimum hopcount paths but its adaptation to more general shortest-path concepts is straightforward.

2.1 Why a distributed heuristic algorithm for service migration

Centralized solutions are inefficient, if at all feasible, for our problem. From a pure algorithmic point of view, the 1-median problem complexity[2] is bounded by $O(|V|^3)$. Hence, while not prohibitive, it does not scale well and improvements are necessary, especially for larger networks. More significantly, centralized approaches assume the existence of a super-entity with global topological and service demand information that has the resources and the mandate to determine and realize the placements. This implies an implicit logical hierarchy in the role of network nodes, which in many cases is not present. Moreover, given that (minor) user demand shifts or network topology changes may be frequent and alter the optimal service location, it is neither practical nor affordable to each time centrally compute a new problem solution.

Our approach is to replace the one-shot placement of service with its few-step migration towards the optimal location. This way we end up solving a few 1-median problems of dramatically smaller scale and complexity instead of coping with the global 1-median optimization problem. Central to the algorithm is a metric inspired from Complex Network Analysis [16] that we call Weighted Conditional Betweenness Centrality (wCBC) [12]. For every transit location of the service in the network, the wCBC is a measure of the demand each node routes towards the current service host node and is used for two tasks. First, it identifies nodes in $\widetilde{G}$ with the highest $wCBC$ values as candidates for hosting the service in the next iteration. These nodes form the service-host-node-dependent 1-median subgraph, wherein the optimization for the next-best service location is solved. Secondly, the metric simplifies the *mapping* of the service demand from the rest of the network nodes on this subgraph. This task is deemed mandatory to appropriately weigh the service demand gradients across the network. We detail the metric and its practical interpretation in Section 3.

3 Weighted Conditional Betweenness Centrality

Central to our *distributed* approach is the Weighted Conditional Betweenness Centrality ($wCBC$) metric. It originates from the well-known betweenness centrality metric and captures both topological and service demand information for each node.

3.1 Capturing network topology: from BC to CBC

Betweenness centrality (BC) reflects to what extent a node lies on the shortest paths linking other nodes. Let σ_{st} denote the number of shortest paths between any two nodes s and t in a connected graph $G = (V, E)$. If $\sigma_{st}(u)$ is the number of shortest paths passing through the node $u \in V$, then the *betweenness centrality* index of node u is given by:

$$BC(u) = \sum_{s,t \in V, s \neq t \neq u} \frac{\sigma_{st}(u)}{\sigma_{st}} \qquad (2)$$

1. For instance, a customized tour service generated by a mashup tool user to provide urban points of a certain interest, would most likely be accessed only by those who share that interest and reside in the same area.

2. In comparison, the k-median problem is NP-hard in general topologies so that much of the research effort around it has been devoted to the design of efficient approximation algorithms [15].

$BC(u)$ captures the ability of a node u to control or assist the establishment of paths between pairs of nodes. It is an average value estimated over all network pairs.

In [17] we proposed the Conditional BC (CBC), as a way to capture the topological centrality of a random network node with respect to a specific node t. It is defined as

$$CBC(u;t) = \sum_{s \in V, u \neq t} \frac{\sigma_{st}(u)}{\sigma_{st}} \quad (3)$$

with $\sigma_{st}(s) = 0$. Note that the summation is over all $|V-1|$ node pairs involving node t rather than all possible $|V|(|V|-1)$ node pairs, as in (2). Effectively, CBC assesses to what extent a node u acts as a *shortest path aggregator* towards the current service location t, by enumerating the shortest paths to t involving u from all other network nodes. In the supplemental material we compute the CBC values and their distributions over simple network topologies.

3.2 Capturing service demand: from CBC to wCBC

A high number of shortest paths through the node u does not necessarily mean that equally high demand load stems from the sources of those paths. *Weighted conditional betweenness centrality* ($wCBC$) enhances the pure topology-aware CBC metric in a way that takes into account the service demand that can be routed through the shortest paths towards the service location [12]. The shortest path ratios of $\sigma_{st}(u)$ to σ_{st} in Eq. (3) are now weighted by the demand loads generated by each node s as follows:

$$wCBC(u;t) = \sum_{s \in V, u \neq t} w(s) \cdot \frac{\sigma_{st}(u)}{\sigma_{st}}. \quad (4)$$

Note that $\sigma_{ut}(u) = \sigma_{ut}$ so that the $wCBC(u;t)$ value of node u is lower bounded by its own demand $w(u)$. Therefore, $wCBC$ assesses to what extent a node can serve as *demand load concentrator* towards a given service location. Clearly, when the demand for a service is uniformly distributed across the network nodes, the $wCBC$ metric degenerates to the CBC one, within a scale constant.

3.2.1 Approximating the metric with measurements

The $wCBC(u;t)$ metric practically represents the service demand that node u routes toward node t, including its own demand $w(u)$ and the transit demand $w_{trans}(u;t)$ flowing from other network nodes through u towards t. Therefore, individual nodes may, in principle, *estimate* their own metric values $w\hat{C}BC$ through passive measurements [18] of the service demand they route towards the current service host node. In other words, what is actually computed theoretically in (4) for node u demanding global information about the network topology and service demand, can be locally approximated by u providing the basis for the practical implementation of a distributed solution. The approximation lies in the fact that what is measured, even with perfect accuracy, is not always equal to the nominal $wCBC$ value, as specified in (4). For example, when there are, say m, shortest paths between a given node pair (s,t) but the routing protocol uses only one of those, a node will measure the full demand of s, whereas, theoretically, the contribution to the nominal $wCBC$ value is the $(1/m)^{th}$ of the measured one. We will see later in section 7 that what matters is the estimate of the *actual* demand routed through the node rather than a $wCBC$ approximation.

4 The cDSM Algorithm Description

Our centrality-driven Distributed Service Migration Algorithm (cDSMA) progressively steers the service towards its optimal location via a finite number of steps.

Step 1: Initialization. The first algorithm iteration is executed at node s in $\widetilde{G}$ that initially generates the service facility (pseudocode $line\ 2$). In subsequent iterations, the new reference node is the one each time hosting the service.

Step 2: Metric computation and 1-median subgraph derivation. Next, the $wCBC(u;s)$ metric is computed[3] for every node u in the network graph $\widetilde{G}$. Nodes in $\widetilde{G}$ featuring the top $\alpha\%$ $wCBC$ values, together with the node currently hosting the service ($Host$) form the 1-median subgraph $\widetilde{G}_{Host}$ over which the 1-median problem will be solved ($lines\ 3-4$ and $15-16$). Clearly, its size and the algorithm complexity are directly affected by the α parameter choice.

Step 3: Mapping the demand of the remaining nodes on the subgraph. To account for the contribution of the "outside world" to the service provisioning cost, the demand for service from nodes in $G \setminus \widetilde{G}_{Host}$ (*i.e.*, the non-shaded nodes in fig. 1.c) is mapped on the $\widetilde{G}_{Host}$ ones. To do this correctly and with no redundancy, the algorithm credits the demand of some outside node z only to the first "entry" $\widetilde{G}_{Host}$ node encountered on each shortest path (over G), from z towards the service host. Thus, the weights $w(n)$ for calculating the service access cost at node n in the $\widetilde{G}_{Host}$ subgraph (see Section 2) are replaced by *effective demands*: $w_{eff}(n;Host) = w(n) + w_{map}(n;Host)$, where (assuming that $Host$ is node t):

$$w_{map}(n;t) = \sum_{z \in \{G \setminus \widetilde{G}_t\}} w(z) \frac{\sigma'_{zt}(n)}{\sigma_{zt}} \quad (5)$$

$$\sigma'_{zt}(n) = \sum_{j=1}^{\sigma_{zt}} \mathbb{I}_{\{n \in SP_{zt}(j) \bigcap n = \underset{u \in SP_{zt}(j)}{\operatorname{argmin}}\ d(z,u)\}}$$

with $SP_{zt}(j)$ standing for the j^{th} element of the shortest path set from node z to node t. For example, in fig. 1.c the original service demand of node, say, 16 is not mapped on all the $\widetilde{G}_7$ nodes lying on the shortest paths from 16 to the $Host$ 7 (*i.e.*, 11, 12 and 8), but only on 11.

The mapping step and its rationale can be better understood in the following example. In the network of fig. 1.d the service migrates towards the lowest cost location, which under uniform demand is node C. Assume that the $\widetilde{G}_{Host}$ subgraph size is 4 and at some migration step the service resides at node F. The top $wCBC$ nodes around F are C, K and L. The cDSMA assigns w_{eff} values only to the entry-nodes C and L, $w_{eff}(C;F) = 6$ and $w_{eff}(L;F) = 3$, and while setting the w_{map} values of nodes F and K to

3. For the actual $wCBC$ computation, which involves solving the all-pairs shortest path problem, we properly modified the scalable algorithm in [19] for *betweenness centrality* computation, with runtime $O(|V||E|)$.

zero, effectively identifies the gradient direction towards the node C. Thus, it better projects the demand attraction forces on the selected nodes, C being the stronger. On the contrary, if we map the demand of nodes in $G \setminus \widetilde{G}_{Host}$ on *all* $\widetilde{G}_{Host}$ nodes, we end up with $w_{eff}(F;F) = 8$ and $w_{eff}(K;F) = 3$. The service then cannot identify node C as the next-best location and locks at node F.

Step 4: 1-median problem solution and service migration to the new host node. Any centralized technique (*e.g.*, [15]) may be used to solve this small-scale optimization problem and determine the next best location of the service in $\widetilde{G}_{Host}$. Note that the pairwise physical distances between $\widetilde{G}_{Host}$ nodes in Eq. (1) are computed over the original graph G (see fig. 2). We call s the current service location ($line$ 21) while the optimal one in $\widetilde{G}_{Host}$ is assigned to $Host$ ($line$ 22). As long as node s (a) yields higher cost than the candidate $Host$ node ($line$ 10); and (b) the candidate $Host$ has not been used as a service host before ($lines$ 11−13), the service is moved there and the algorithm iterates through steps 2-4. Progressively, cDSMA steers the service to the (globally) lowest-cost location.

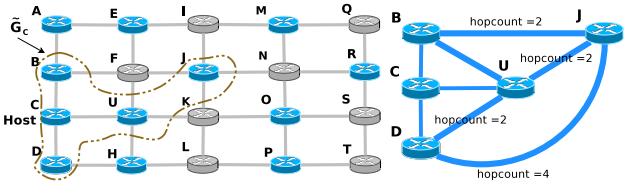


Fig. 2. **Left**: Selected 1-median subgraph $\widetilde{G}_C$ comprised solely of (highlighted) nodes that represent service host candidates over a non-weighted grid topology. **Right**: The corresponding overlay that reflects the physical distances between all $\widetilde{G}_C$ node pairs. Aggregated physical links are presented with thick lines.

Algorithm 1 cDSMA in $\widetilde{G}(V_S, E_S)$

1. *choose randomly node* s
2. *place* $SERVICE$ @ s
3. **for all** $u \in \widetilde{G}$ **do** *compute* $wCBC(u;s)$, *set* $flag(u) = 0$
4. $\widetilde{G}_s \leftarrow \{\alpha\%\ of\ \widetilde{G}\ with\ top\ wCBC\ values\} \cup \{s\}$
5. **for all** $u \in \widetilde{G}_s$ **do**
6. *compute* $w_{map}(u;s)$
7. $w_{eff}(u;s) \leftarrow w_{map}(u;s) + w(u)$
8. *compute cost* $C(u)$ *in* $\widetilde{G}_s$
9. $Host \leftarrow$ *1-median solution in* $\widetilde{G}_s$
10. **while** $C_{Host} < C_s$ **do**
11. **if** $flag(Host) == 1$ **then**
12. *abort*
13. **else**
14. *move* $SERVICE$ *to* $Host$, $flag(s) = 1$
15. **for all** $u \in \widetilde{G}$ **do** *compute* $wCBC(u;Host)$
16. $\widetilde{G}_{Host} \leftarrow \{\alpha\%\ of\ \widetilde{G}\ with\ top\ wCBC\ values\} \cup \{Host\}$
17. **for all** $u \in \widetilde{G}_{Host}$ **do**
18. *compute* $w_{map}(u;Host)$
19. $w_{eff}(u;Host) \leftarrow w_{map}(u;Host) + w(u)$
20. *compute cost* $C(u)$ *in* $\widetilde{G}_{Host}$
21. $s \leftarrow Host$
22. $Host \leftarrow$ *1-median solution in* $\widetilde{G}_{Host}$
23. **end if**
24. **end while**

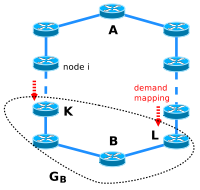


Fig. 3. A ring topology of $N = 2k$ nodes under a non-uniform demand pattern results in symmetric (with respect to B) demand mapping on the G_B entry nodes K and L. This blocks the service migration process and yields a highly suboptimal solution.

Convergence and approximation properties: We complete the description of cDSMA by elaborating on its convergence properties and theoretic capability to approximate the optimal solution. Clearly, a service facility following the migration process of Algorithm 1 will visit any $\widetilde{G}$ network node at most once (see condition in line 12). Thus, our heuristic will take $O(|V|)$ steps to terminate. Its theoretical performance bounds are studied next.

Proposition 4.1: cDSMA provides no constant factor approximation guarantee.

Proof: We sketch a counterexample that leads to arbitrarily bad solution quality: Assume, without loss of generality, that a ring topology consists of $N = 2k$ ($k \in \mathbb{Z}^+$) potential service host nodes *i.e.*, $\widetilde{G} \equiv G$. Every node but one aggregates a unit of demand load from the users it serves (see fig. 3); the single heavy hitter node A generates W units of demand load and the service facility is generated at the anti-diametric ring node B. Under cDSMA the current service host B will select αN nodes with $\alpha < 1$, that will form its G_B subgraph. Interestingly enough, the demand that will be mapped on the G_B-chain entry nodes (*i.e.*, K and L) is such that the initial location becomes a local minimum for every value of $\alpha < 1$. Therefore, the service remains with node B without initiating at all the migration process. The global access cost $C_{cDSMA}(B)$ is

$$C_{cDSMA}(B) = 2\sum_{i=1}^{k-1} i + Wk = \frac{N^2 + 2N(W-1)}{4} \quad (6)$$

On the other hand, the optimal service location is at node A, where the cost is:

$$C_{OPT} = 2\sum_{i=1}^{i=k-1} i + k = \frac{N^2}{4} \quad (7)$$

Therefore, their ratio equals:

$$\frac{C_{cDSMA}(B)}{C_{OPT}} = 1 + 2\frac{W-1}{N} \quad (8)$$

Eq. 8 shows that the resulting placement may become arbitrary bad as the demand of the heavy hitter rises. □

Proposition 4.1 suggests that there are combinations of network topology and demand that may generate such *symmetric* 1-median subgraph mappings that trap the service in a (local) minimum and prematurely terminate the migration process. On the positive side of the particular unfavorable example, the approximation ratio improves fast as the net-

work size N grows, *i.e.*, when the migration becomes more relevant; whereas for small networks (of one or two tens of nodes), the placement task is computationally feasible centrally. More generally, our experimentation results in Section 6 and, more notably, in Section 8, demonstrate that cDSMA *actually* provides excellent accuracy for all realistic scenarios of network topologies and demand distributions even for very small sizes of the 1-median subgraph. Similar behavior is not uncommon to appear in approximation algorithms, especially those that tackle computationally difficult problems; parallels can be drawn with the well-known k-means algorithm that is a hill-climbing approach to the partitioning of data points into k disjoint clusters. It is widely used in practice although it occasionally converges to local minima that can be arbitrarily bad in terms of accuracy, when compared to the optimal clustering [20].

5 Evaluation Methodology

Our evaluation of the algorithm proceeds in two steps. First, in Section 6, we study its behavior under the ideal assumption that nodes avail perfect global information about the network topology and the service demand. This proof-of-concept validation is carried out over both synthetic and real-world network topologies and under various service demand distributions. Later in Section 8 we assess a realistic protocol implementation proposal, earlier presented in Section 7. Hereafter, we treat each network node as a potential service-providing location *i.e.*, $\widetilde{G} \equiv G$, testing the performance of our algorithm as well as its practical implementations against the worst-case[4] network conditions.

***Network topology**:* The synthetic topologies we experiment with are Barabási-Albert (B-A) graphs [21] and two-dimensional rectangular grids. The specific graph models were chosen deliberately since they bear very different and distinct structural properties. The B-A graphs form pure probabilistically and can reproduce a highly skewed node degree distribution that approximates the power-law shape reported in literature [22]. Grids, on the other hand, exhibit strictly regular structure with constant node degree and diameter that grows exponentially with the number of network nodes. The synthetic network topologies let us highlight the behavior of cDSMA in the presence of general network structure properties.

Nevertheless, the ultimate assessment of our algorithm is carried out over real-world ISP network topologies. The recently published dataset [23] we consider includes numerous snapshots of 14 different AS topologies, corresponding to Tier-1, Transit and Stub ISPs [24]. The data were collected daily during the period 2004-08 with the help of a multicast discovering tool called `mrinfo`, which circumvents the complexity and inaccuracy of more conventional measurement tools such as `traceroute`. We focus on the larger Transit and Tier-1 ISP datafiles sizing up to approximately 1000 nodes and show results for a representative subset featuring adequate variance in size, diameter, and connectivity degree statistics.

***Service demand distribution**:* At a first level, our assessment distinguishes between uniform and non-uniform demand scenarios. Though far from realistic, uniform demand scenarios let us study the *exclusive* impact of network topology upon the behavior of the algorithm. On the contrary, under non-uniform demand distributions, the algorithm is exposed to the *simultaneous* influence of network topology *and* service demand dynamics. Mathematically speaking, a Zipf distribution models the preference $w(n; s, N)$ of nodes $n, n \in \mathcal{N}$ to a given service as: $w(n; s, N) = \frac{1/n^s}{\sum_{l=1}^{N} 1/l^s}$. Practically, the distribution could correspond to the normalized service request rate. Increasing the parameter s from 0 to ∞, the distribution asymmetry grows from zero (uniform demand) towards higher values. At a second level, we consider two options as to how the non-uniform service demand emerges spatially within the network. In the default option, each node randomly generates demand according to the Zipf law. The alternative is to introduce geographical correlation by concentrating nodes with high demand in the same network area. This second scenario lends itself to modeling services with strongly local scope.

***Algorithm performance metrics**:* We are concerned with two metrics when assessing the performance of cDSMA. The first one relates to its accuracy, *i.e.*, how well it approximates the optimal solution. It is defined as the *average normalized excess cost*, β_{alg}, and equals the ratio of the service access cost our algorithm achieves, $C_{alg}(\alpha; G, \overline{w})$, over the cost achieved by the optimal solution (derived by a brute-force centralized algorithm assuming availability of global topology and demand information) $C_{opt}(G, \overline{w})$, for given network topology G and demand distribution $\overline{w}$:

$$\beta_{alg}(\alpha; G, \overline{w}) = E[\ \frac{C_{alg}(\alpha; G, \overline{w})}{C_{opt}(G, \overline{w})}\ ] \qquad (9)$$

β_{alg} clearly depends on the percentage α of the network nodes participating in the solution. Less intuitively yet inline with Section 4, greater α values may not result in β_{alg} improvement. Closely related to β_{alg} are the $\lceil |G_{Host}| \rceil_{\epsilon}$ indices corresponding to the minimum size of the 1-median subgraph our heuristic requires to achieve access cost that falls within $100 \cdot \epsilon\%$ of the optimal. Hereafter, we set ϵ to 0.025 and use the notation without the subscript. The second metric is the *migration hop count*, h_m, which is generally a function of α and reflects how fast the algorithm converges to its (sub)optimal solution. Smaller h_m values imply faster service deployment and less overhead involved in transport and service set-up/shut-down tasks.

When the involved parameters vary along the network instances (e.g., B-A graphs) or the service demand distribution, we present results that are the averages over 10 different topologies and/or 10 different vectors of Zipf-distributed demand values. We repeat a number of simulation runs (*i.e.*, 20, 40 and 50) to obtain a sample up to at least 7% of the {net topology, demand vectors} space. Typically, the average values are presented together with the 95% confidence intervals, estimated over the runs.

4. For instance, a limited number of service host nodes may allow the use of unicast routing rather than some sort of flooding scheme for communicating the protocol-related messages of a cDSMA implementation.

6 cDSMA Proof-of-concept study

6.1 Synthetic topologies experiments

Figure 4 plots the average normalized excess cost β_{alg} for B-A like graphs[5] and grids of 100 nodes against the G_{Host} size under different demand patterns. As expected, the error induced by cDSMA tends to decrease with the G_{Host} size despite the non-monotonicity of the corresponding curves.

Grid topology: Under uniform demand the regular structure of the grid renders the most central points of the grid, *i.e.*, its barycenter, the optimal service locations. Therefore, the demand gradient is directed towards the grid center. The migrating service moves along it; nevertheless, the closer it gets to the grid center the more intense are the attraction forces from the nodes that lie behind it. As a result, there are nodes around the optimal grid location(s) that impede the local search and become *traps* for the migrating service (see fig 5.a). This is due to a combination of the subgraph size (*i.e.*, α) and the topology symmetry. More specifically, the demand mapping step correctly assigns higher w_{eff} values to those nodes in the G_{Host} subgraph that are closer to the optimal location, pulling the service to its direction. However, the current host exhibits the lowest access cost among all the G_{Host} nodes and terminates the service migration. As the α percentage grows bigger the optimal G_{Host} location is shifted away from the current service host and thus, the number of trap nodes decreases (see fig 5.b,c).

The shift from uniform to a more skewed spatially uncorrelated demand distribution results in the trap nodes being anywhere within the network. In particular, the traps will appear in nodes that lie somewhere in-between heavy hitters and stand under the influence of approximately equal attraction forces (fig 5.d). The service gets locked there, much as happened under the uniform demand case. The difference now is that as we let α grow, the G_{Host} subgraph will stretch to many directions, namely the ones that heavy hitters use to reach the current service host. This means that we will need on average more nodes than before in order to overcome the traps. Consequently the normalized cost ratio converges to one more slowly (fig. 4.d).

When the demand distribution is spatially correlated (*i.e.*, the interest in the service is concentrated in a particular neighborhood, as when the service has strongly local scope), a cluster of nodes with high service demand appears in a random area within the grid. Let K cluster nodes collectively represent some percentage $z\%$ of the total demand for the service, whereas the other $N-K$ nodes share the remaining $(100-z)\%$ of the demand. We call the ratio $z/(100-z)$ the demand *spatial contrast* C_{sp}. In 2D grids, clusters are formed by a cluster head node together with its R-hop neighbors. The contrast can then be written as: $C_{sp}(R,s) = \frac{\sum_{n=1}^{K} w(n;s,N)}{\sum_{n=K+1}^{N} w(n;s,N)} = \frac{\sum_{n=1}^{K} 1/n^s}{\sum_{n=K+1}^{N} 1/n^s}$ and the average normalized excess cost becomes a function of both α and the contrast value. The $\beta_{alg}(\alpha, C_{sp})$ values for a 10x10 grid topology under spatially random and correlated ($R=1$) distribution of demands are reported in the rightmost columns of Table 1, respectively. Having the top demand values stemming from a certain network neighborhood we actually "produce" a single pole of strong attraction for the migrating service. cDSMA now follows the demand gradient more effectively than before. As the percentage of the total cluster nodes' demand grows larger (*i.e.*, higher C_{sp}), the pole gets even stronger driving the service firmly to the optimal location.

TABLE 1
Impact of spatially correlated service demands

skewness s	$C_{sp}(1,s)$	$\beta_{alg}(0.1)$	$\beta_{alg}(0.1, C_{sp})$
1	0.786	1.026±0.016	1.013± 0.010
2	8.540	1.002±0.003	1.0±0.0

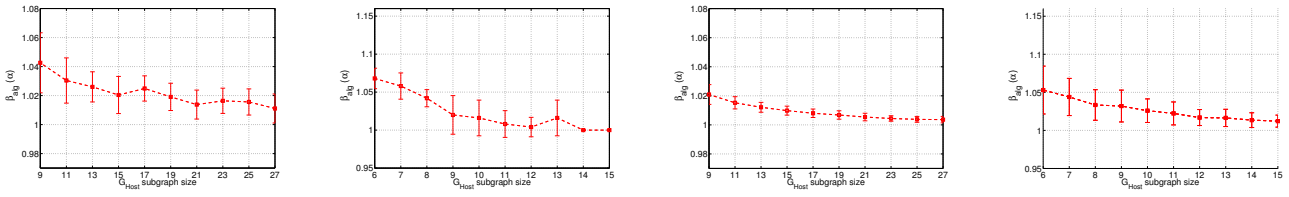
B-A graphs: The B-A graph characteristics seem to amplify the service trap phenomena. Regardless the generation location the high-degree hub nodes of B-A graphs [21] are correctly identified as low-cost solutions and therefore cDSMA has the service moved there already with its first hop. As the hub node communicates directly with almost all G_{Host} nodes, it is likely to become the minimum cost node. In a B-A graph of 100 nodes, where we iterate generating a service at each node under uniform demand, 62 times the service locks on 3 different hub nodes other than the optimal; and this is almost consistently done in the first hop. A closer look reveals that such local minima appear robust to the G_{Host} size in line with fig 4.a. Fig. 5.f depicts an example of cDSMA behavior when the service is generated in some node and subsequently moves to a sub-optimal hub. By increasing the number of selected nodes we essentially tend to include in the G_{Host} subgraph a direct neighbor of the current high-degree service host. Thus we get the same final cost across a wide α range. Moreover, when the G_{Host} is big enough to find the optimal host, it does so by directly moving there from the generation location; accordingly, the hopcount remains on average slightly over one. When a non-uniform demand pattern emerges, the combination of the high degree nodes with the presence of heavy-hitters, on average works in favor of cDSMA. In the above B-A graph of 100 nodes, for 39.5 times averaged over 4 different demand vectors, the service locks on the 3 different sub-optimal hub nodes.

The above results suggest that the cDSMA performance while closely approximating the optimal exhibits sensitivity to certain connectivity properties of the network topology. In the presence of high degree hub nodes assisted by low average path length the algorithm requires relatively high G_{Host} size to correctly determine the next best solution. Moreover, the randomization introduced by skewed demand distributions does not necessarily benefits the algorithm (*e.g.*, grids). In the sequence, we validate these general rules about cDSMA over real-world network topologies.

6.2 Real-world network topologies experiments

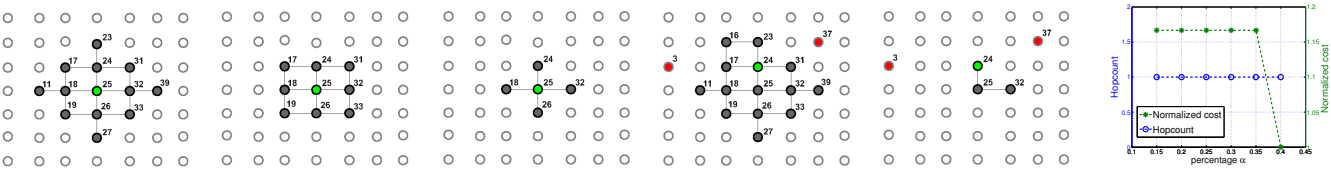
The ultimate assessment of cDSMA is carried out over real-world ISP network topologies, which do not typically have the predictable structural properties of B-A graphs and grids. Still, as detailed in the supplement, insightful analogies regarding the cDSMA behavior can be drawn between

5. To obtain proper B-A networks, we would need network sizes in the order of thousands of nodes. Hence, strictly speaking, the scale-free networks of a few hundred nodes' size we experiment with are small to be called B-A networks. We retain the name for ease of reference.



a. B-A (s=0) b. 10x10 Grid (s=0) c. B-A (s=1) d. 10x10 Grid (s=1)

Fig. 4. Synthetic topologies of 100 nodes : cDSMA accuracy *vs.* 1-median subgraph size under uniform (a,b) and Zipf (c,d) demand distribution.



a. $|G_{Host}| = 5$ b. $|G_{Host}| = 7$ c. $|G_{Host}| = 12$ d. $|G_{Host}| = 5$ e. $|G_{Host}| = 16$ f. B-A Traps Robustness

Fig. 5. The trap nodes (dark dots) as a function of the G_{Host} subgraph size for a 7x7 grid under uniform (a,b,c) and non-uniform demand (d,e) that emerges by nodes 3 and 37 being equally heavy hitters. For the former case (optimal is 25) the traps vanish when $|G_{Host}| = 14$, while for the latter (optimal is 24) when $|G_{Host}| = 18$. In subfigure f, scaling the G_{Host} size can hardly help the service overcome a high degree B-A trap node.

real-world and synthetic topologies. Table 2 summarizes the performance of cDSMA over the data that represent the real-world topologies[6]. It reports the minimum number of nodes $|G_{Host}|$ required (across the demand vectors in case of non-uniform demand pattern) to achieve a solution that lies within 2.5% of the optimal; the corresponding average migration hop count h_m is also shown.

The $|G_{Host}|$ values show a notable insensitivity to both topological structure and service demand dynamics. Although the considered ISP topologies differ significantly in size and diameter, the required 1-median subgraph size does not change substantially. Employing 4.5% of the total number of nodes or 6% for the least favorable case suffices to obtain very good accuracy across all ISP topologies. Likewise, the required 1-median subgraph size remains in almost all experiments practically invariable with the demand distribution skewness. Although for larger values of s, few nodes become stronger attractors for the algorithm, the added value for its accuracy is in most considered datasets negligible. Even for the larger topologies that appear more sensitive to service demand variations, the $|G_{Host}|$ differences across the skewness values are no more than 4% of the total network size. This two-way insensitivity of cDSMA is of major importance as: a) the computational complexity of the local 1-median problem can be negligible and scales well with the network size and diameter. b) the algorithm performance is robust to possibly inaccurate estimates of the service demand each node poses. In Section 8 we will see how our practical cDSMA implementation maintains similar welcome characteristics.

6.3 cDSMA vs. locality-oriented service migration

It is instructive to compare cDSMA against stricter "local-search" approaches to distributed service migration. One example is the *R*-ball heuristic used in [26], where the search for a better service host is a priori bounded within the r-hop distance neighborhood of the node each time hosting the service[7]. On the contrary, cDSMA invests more effort and intelligence in selecting the service host candidates. The resulting 1-median subgraph is spatially stretched along paths consisting of highly "central" nodes and oversteps the local neighborhood "barriers". This is clearly illustrated in fig. 6 showing the hopcount distribution between the service host node and the selected each time 1-median subgraph nodes, as extracted from five executions of cDSMA under Zipf demand with s=2.

To highlight what cDSMA gains by the more informed derivation of the 1-median subgraph, we compare it against its variant that implements the R-ball heuristic, hereafter called Locality-Oriented Migration (LOM). Both variants use the same demand mapping mechanism (Section 4) to capture the demand from nodes lying outside the induced 1-median subgraphs. The comparison between cDSMA and LOM, illustrated in Table 3, proceeds as follows. We first generate asymmetric service demand (Zipf distribution with $s = 1$) across the network. We compute the globally optimal service host node and select a fixed set of service generation nodes, at D_{gen} hops away from the optimal location. We then calculate the values of β_{alg} and h_m metrics for the two approaches along with the mean number of nodes included in the 1-median subgraph G_{LOM} for each execution. For the cDSMA, we have set the parameter

6. Several files miss some edges resulting in more than one connected components [24]. Thus, a pre-processing task using a linear-time algorithm [25], is needed to retrieve the maximal connected component mCC.

7. A direct side-by-side comparison of the two service migration approaches is not applicable since the r-ball heuristic is combined with service replication, thus coping with the k-median problem.

TABLE 2
Mean hopcount and ceiling $|G_{Host}|$ values for various datasets under different demand distributions

					s=0		s=1		s=2	
ISP	Dataset id/AS#	mCC nodes	Diameter	<Degree>	$< h_m >$	$\|\|G_{Host}\|\|$	$< h_m >$	$\|\|G_{Host}\|\|$	$< h_m >$	$\|\|G_{Host}\|\|$
*type:***Tier-1**										
Global Crossing	36/3549	76	10	3.71	1.00±0.23	6	1.63±0.35	3	3.32±0.78	2
-//-	35/3549	100	9	3.78	1.30±0.34	7	1.26±0.16	6	1.45±0.22	4
NTTC-Gin	33/2914	180	11	3.53	1.0±0.0	18	1.11±0.13	10	1.08±0.09	8
Sprint	23/1239	184	13	3.06	1.40±0.36	8	1.22±0.15	7	1.95±0.31	4
-//-	21/1239	216	12	3.07	1.40±0.41	7	1.42±0.19	6	1.50±0.12	5
Level-3	27/3356	339	24	3.98	2.23±0.58	4	3.15±0.24	3	2.41±0.40	5
-//-	13/3356	378	25	4.49	2.27±0.59	4	2.48±0.37	4	2.22±0.34	6
Sprint	20/1239	528	16	3.13	1.40±0.62	11	1.27±0.21	21	1.09±0.11	24
*type:***Transit**										
TDC	52/3292	72	9	3.28	1.20±0.29	5	1.09±0.12	5	1.46±0.28	4
DFN-IPX-Win	41/680	253	14	2.62	1.40±0.36	7	1.35±0.23	6	1.49±0.19	6
JanetUK	40/786	336	14	2.69	1.23±0.31	11	1.13±0.08	9	1.30±0.12	6
Iunet	39/1267	711	13	3.45	1.0±0.0	11	1.03±0.06	43	0.99±0.01	9

$\alpha = 2\%$ yielding 1-median subgraphs of size 4 for Datasets 23, 33 and 7 for Dataset 27 (Table 3).

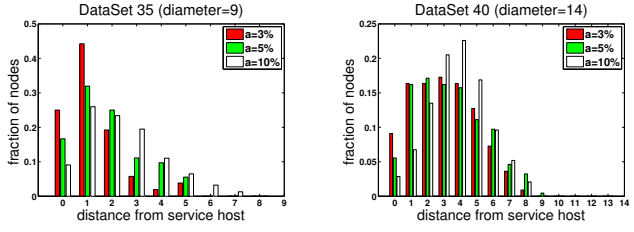


Fig. 6. Hopcount distribution of 1-median nodes from the Host under cDSMA.

In most of our experiments, LOM demonstrates comparable accuracy to our heuristic. It suffices to set R=1 to obtain near-optimal service placements, while taking more nodes into account seems to offer no extra gain. Still, the LOM approach exhibits two disadvantages when compared against cDSMA. First, LOM needs far more migration hops than cDSMA since it hard bounds the length of a single migration hop. For R=1, in particular, the LOM performs as many hops as D_{gen} to reach the optimal service location. On the other hand, cDSMA chooses the most "appropriate" candidate host nodes allowing them to stretch across the demand gradient direction; consequently, it leads the service fast to prominent locations. As a positive side-effect, it almost decouples the convergence speed of the algorithm from the service generation location. Thus, it does not differentiate user nodes according to their proximity to the globally optimum location inducing a notion of fairness in the performance they get. Secondly, the LOM heuristic imposes much less flexibility in determining the size of the 1-median subgraph. For R=2, LOM may end up seeking for the next best service host node among an order of magnitude more candidate hosts than cDSMA. On the other hand, taking R=1 does not always suffice; in the Dataset 27 experiment ($D_{gen} = 14$) the migration process stops prematurely one hop away from the service generation location yielding prohibitively high cost.

7 A PRACTICAL cDSMA IMPLEMENTATION

A practical real-world implementation of cDSMA needs to cope with two main challenges. Firstly, information residing with individual network nodes has to be collected at the node each time hosting the service. This information includes the $wCBC$ and w_{eff} values that guide the 1-median subgraph derivation and the demand mapping steps, respectively (Section 4). Secondly, even if this information is compiled by each node in a distributed manner, equations 4 and 5 imply that global information about the network topology and demand is required.

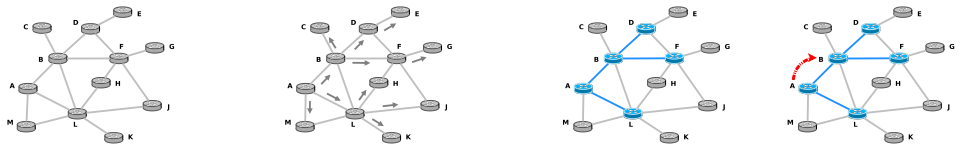
The second concern is partly addressed by the net interpretation of the $wCBC$ metric, as already discussed in section 3.2.1. The $wCBC(u;t)$ value represents the aggregate traffic demand flowing through node u towards the service host node t. Therefore, the $\{wCBC\}$ values can be locally inferred at each node by directly measuring their transit traffic load that is destined for node t. How accurately the measured values $\{w\hat{CBC}\}$ match the theoretical values $\{wCBC\}$, as defined in eq. 4, depends on the network topology and routing protocol. The network topology may present each single node pair with one or more shortest (a.k.a. minimum hopcount) paths; while the routing protocol may use one, more than one or none of them. Therefore, leaving measurement inaccuracies aside, the two sets of values coincide when: (a) the topology gives rise to a single shortest path between all network node pairs (*e.g.*, tree topologies) and the routing protocol routes traffic over this single shortest path; (b) the topology induces multiple shortest paths between all network node pairs (*e.g.*, lattice topologies) and the routing protocol splits the traffic demand equally among all of them. In general, the routing protocol can be viewed as a transformation $\mathcal{F} : \{wCBC\} \rightarrow \{w\hat{CBC}\}$, which becomes an identity one in the two particular scenarios mentioned above. In this section, we present the proposed cDSMA implementation in a step-by-step fashion. We iterate on how this implementation addresses the aforementioned practical challenges under a *single-path routing* (SP) hypothesis (see fig.7). Our comprehensive study of the *multi-path routing* case (MP) is presented in the supplemental material due to limited space.

7.1 Service host advertisement

Every time the service carries out a cDSMA-driven migration hop, the new host initiates a service advertisement phase (see Fig. 7.b) to inform all network nodes about the current service location. This task may be carried out by any efficient flooding scheme requiring $O(|E|)$ messages and $O(D)$ time, where D is the network diameter. Note that this step is *sine qua non* for all protocol instances real-

TABLE 3
Convergence speed and accuracy of LOM and cDSMA on real-world topologies

		Dataset 23				Dataset 33					Dataset 27					
	D_{gen}:	3	4	5	7	3	4	5	7	10	3	4	5	7	10	14
LOM	β_{alg}	1	1.0299	1.0434	1.0299	1	1	1	1	1	1	1	1	1	1	2.67
R=1	h_m	3	2	2	4	3	4	5	7	10	3	4	5	7	10	1
	$mean\|G_{LOM}\|$	7.2	9.5	8	7.8	5	5	5.3	5.9	5.8	5	4.8	4.3	5.2	5	5.5
LOM	β_{alg}	1	1.0299	1.0434	1.0299	1	1	1	1	1	1	1	1	1	1	1
R=2	h_m	2	1	1	2	2	2	3	4	5	2	2	3	4	5	8
	$mean\|G_{LOM}\|$	28.3	29.5	28	22.3	14.3	14.7	14.3	18.2	18.8	17.7	14.7	11	15.2	15	14.7
cDSMA	$\beta_{alg}(2\%)$	1	1.0299	1.0434	1.0299	1	1	1	1	1	1	1	1	1	1	1
	h_m	1	1	1	2	1	2	2	3	4	1	1	1	2	2	2



a. The graph $G(E,V)$ b. Advertisement phase (service at A) c. Host determines G_A d. Host solves 1-median

Fig. 7. Example of cDSMA protocol implementation under single-path routing and uniform demand.

izing distributed service placement under dynamic demand patterns (*e.g.*, [26], [27]).

7.2 Reporting of local wCBC estimates and inference of the 1-median subgraph

Drawing on passive measurements [18] each node $u \in G = (V, E)$ derives an estimate $wC\hat{B}C(u;t)$ of the traffic demand flowing through u towards the current service host node t. These estimates are then reported to node t via *dedicated measurement-reporting* messages in $O(D)$ time. Nodes can separately report the portion of traffic demand $w(u)$ originating from themselves and the transit traffic $w_{trans}(u;t)$ originating from other nodes towards t, with

$$wC\hat{B}C(u;t) = w_{trans}(u;t) + w(u) \qquad (10)$$

The service host node then ranks the reported values and selects the nodes with the top $\alpha|V|$ values to form the 1-median subgraph, wherein the 1-median problem will eventually be solved (see 3.2.1).

Besides bearing the two traffic demand values, these $O(|V|)$ dedicated messages are exploited further to offer the current service host node with a (partial) view of the 1-median subgraph topology. As each measurement-reporting message travels on its shortest path towards the $Host$, it records all nodes lying on it. The inferred G_{Host} topology exhibits attributes that depend on whether the employed routing protocol makes use of one or more, when available, shortest paths among given node pairs; herein we present our study for the former case. Single-path routing is the standard practice; a single path is used for routing traffic between two nodes at any point in time and resilience to failures is achieved by the use of (hot) stand-by paths (links) that are activated upon failure of the operational one. Under the SP routing policy the following proposition is relevant:

Proposition 7.1: In each iteration of cDSMA, the 1-median subgraph under single-path routing policy is a tree rooted at the current service host node.

Proof: Since each node communicates with the current service host $Host$ via a single shortest-path route, the selection of nodes for the 1-median subgraph is carried out on the *spanning tree* rooted at $Host$, as induced by the routing protocol operation. The only case that the resulting subgraph G_{Host} may not be a tree is when the node selection criterion results in a non-connected subgraph. But this is not possible with $wC\hat{B}C$ as the node selection criterion. To see this, consider any node, say z, which is part of the 1-median subgraph. Then for all nodes k lying on the $A - Host$ shortest path, it holds

$$\begin{aligned} wC\hat{B}C(k;Host) &= w_{trans}(k;t) + w(k) \\ &\geq w_{trans}(z;t) + w(z) + w(k) \\ &= wC\hat{B}C(z;Host) + w(k) \\ &\geq wC\hat{B}C(z;Host) \end{aligned} \qquad (11)$$

that is, all nodes lying in the shortest path $A - Host$ report $wC\hat{B}C$ values at least as high as that of node A. Hence, if A is selected as member of the subgraph, all nodes between A and $Host$ on the tree branch $A - Host$ are selected as subgraph nodes as well, implying that the subgraph is connected and, thus, a tree [8]. □

Corollary 7.1: Under SP routing, the distance of any G_{Host} node from the $Host$ is upper-bounded by $\alpha|V|$-1.

Proof: Since G_{Host} is a tree rooted at $Host$, the greatest distance from the root to a node equals the height of an $\alpha|V|$-node tree, which is $a \cdot |V| - 1$. □

We denote the set of node records appearing on the message of node x with msg_x, their cardinality with $|msg_x|$ and the m^{th} node entry in msg_x with $msg_x(m)$. Fig. 8 (left) presents the topological information that would become available to the node A in Fig. 7.c by the end of this step.

Next we explain that the topological information communicated by these dedicated messages suffices for carrying out the demand mapping task on the 1-median subgraph *without the need for any additional feedback from the*

8. Even when there is a tie in the $wCBC$ values of two or more nodes, the current service host can use the topological information in the measurement-reporting messages to choose the node(s) that preserve the tree property.

network nodes. In the supplementary material we show how this can also be realized under the MP routing option.

7.3 Demand mapping on the 1-median subgraph

After the derivation of the 1-median subgraph, the current service host needs to further process the $\alpha|V|$ measurement-reporting messages that correspond to the selected subgraph nodes.

Proposition 7.1 has two favorable implications that greatly simplify the realization of the demand mapping task. Firstly, the w_{eff} values of all nodes can be computed through direct additions and subtractions of the reported $wC\hat{B}C$ values. Therefore, for leaf nodes x of the resulting tree T, $w_{eff}(x;t) = wC\hat{B}C(x;t)$; whereas, for all inner nodes u

$$w_{eff}(u;t) = wC\hat{B}C(u;t) - \sum_{z \in Ch(u;T)} wC\hat{B}C(z;t) \quad (12)$$

where $Ch(u;T)$ is the set of child nodes of u in T. Secondly, the parsing of the measurement-reporting messages does not have to be exhaustive. Instead, the $\alpha|V|$ messages of the 1-median subgraph (tree) nodes can be sorted and parsed in decreasing length order. Since messages originating from internal tree nodes with a single child are subsets of the longer messages originating from the external tree nodes, they can be safely discarded without any information loss. For example, in fig. 8 (right), the messages msg_F and msg_B are discarded upon parsing their first node entry.

Algorithm 2 Message header parsing and demand mapping under SP (DeMaSP)

1. ***input:*** *set of selected nodes in* G_{Host},
2. $\{msg_u\}\ \forall u \in G_{Host}$
3. ***output:*** *vector* $w_{eff}(u)\ \forall u \in G_{Host}$
4. *Initialization*
5. **for all** $x \in G_{Host}$ **do** $w_{eff}(x) = wC\hat{B}C(x)$
6. *vector* $B \leftarrow$ *sort all* msg_x *in decreasing order of* $|msg_x|$
7.
8. **for** $i = 1$ **up to** $Len(B)$ **do**
9. *parse* $B(i) = msg_x$
10. **for** $m = 1$ **up to** $|msg_x| - 1$ **do**
11. **if** $msg_x(m)$ *is marked* **then**
12. **if** $m > 1$ **then**
13. $k = msg_x(m-1),\ l = msg_x(m)$
14. $w_{eff}(l) = w_{eff}(l) - wC\hat{B}C(k)$
15. **end**
16. *drop* msg_x
17. **else**
18. **if** $m > 1$
19. $k = msg_x(m-1),\ l = msg_x(m)$
20. $w_{eff}(l) = wC\hat{B}C(l) - wC\hat{B}C(k)$
21. **end**
22. *mark* msg_x *as read*
23. **end if**
24. **end for**
25. **end for**
26. $w_{eff}(Host) = w(Host) + (w_{trans}(Host;Host) - \sum_{z \in Ch(Host;T)} wC\hat{B}C(z)$

Algorithm 2, hereafter called DeMaSP, summarizes the process. DeMaSP sequentially parses the measurement-reporting messages of nodes selected for the 1-median subgraph (*selected* nodes) in decreasing length order of their msg part. The output variable w_{eff}, one for each selected node, is initialized to the reported measured traffic values. While parsing the messages, DeMaSP subtracts the portion of traffic demand that has already been credited to nodes of higher depth in the 1-median tree. For instance, it computes the $w_{eff}(B)$ of the internal node B (fig. 7c) by subtracting the sum of the $wC\hat{B}C$ reported values of its child nodes over the emerging tree *i.e.*, D and F, from the node's own $wC\hat{B}C(B;A)$; the outcome equals the $w(C)$ plus the native $w(B)$. Finally, the $Host$ assigns to itself the amount of demand that has not been credited on the selected nodes; it equals its own demand $w(Host)$ plus the difference between the measured incoming traffic demand, denoted by $w_{trans}(Host;Host)$, and the sum of the $wC\hat{B}C$ values of its first neighbours that belong to the tree T of selected nodes *i.e.*, $Ch(Host;T)$.

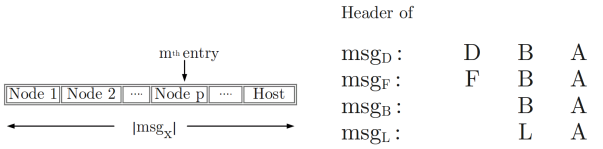


Fig. 8. **Left:** Header format of msg_i^{node1}, the message sent from node 1 of G_{Host} to the $Host$. **Right:** Message headers in decreasing length for the G_A of fig.7.c

7.4 1-median solution within the G_{Host} subgraph

The second input needed for the reduced 1-median solution is the pairwise distances between all G_{Host} nodes. This can be acquired as follows: The current $Host$ notifies each of the top $wC\hat{B}C$ nodes with unicast messages of which other nodes (co-players) are included in the G_{Host} and queries them for their pairwise distances. Each node determines its distance to the other co-players via a mechanism such as the `ping` utility ($O(\alpha^2|V|^2)$ steps, $O(\alpha^2|V|^2)$ messages), and communicates them ($O(\alpha|V|)$ messages) to the host.

7.5 Additional considerations

cDSMA can be applied under any deployed routing strategy, even when the paths actually used are not the minimum hopcount (or shortest in a different sense). In that case, the theoretical $\{wCBC\}$ and the measured $\{wC\hat{B}C\}$ values will deviate, even under single-path routing and tree topologies. The cDSMA will derive a probably different subgraph and the weights of the nodes in this graph will be correspondingly affected. However, the algorithm will carry its task as usual; it cannot know and does not care about learning how exactly the routing protocol in operation transforms the field of theoretical $\{wCBC\}$ values. Similarly, a cDSMA implementation can cope with the limitation of the service hosts to a certain portion of the network nodes, as considered earlier in the algorithm description. Clearly, the 1-median subgraph $\widetilde{G}_{Host}$ under (any sense of) SP routing will not necessarily be connected due to the arbitrary presence of non-service-host nodes across the topology (*e.g.*, Proposition 7.1 does not hold). Nevertheless, its adaptation is straightforward; these nodes should be treated as the non-selected nodes of the DeMaMP algorithm presented in the supplemental material.

8 Performance of cDSMA practical implementations

We now compare both our practical implementations of cDSMA, *i.e.*, over single-path ($cDSMA_{SP}$) and multipath ($cDSMA_{MP}$) routing, against their theoretical primitive ($cDSMA_{TH}$) in Section 4. Effectively, we repeat the experiments of Section 6.2 over the ISP network topologies: the service facilities are generated at the same initial locations, the demand vectors coincide with those employed in Section 6.2, and the 1-median subgraph sizes are set to those $\lceil |G_{Host}| \rceil$ values that yield solutions of cost within 2.5% of the optimal for the theoretical cDSMA (Table 2). For the multipath routing case, we assume that traffic destined for the $Host$ is equally split over each of those u's outgoing links that leverage shortest paths to the $Host$.

We first evaluate our practical implementation assuming that *all* network nodes are engaged in performing and reporting traffic demand measurements (see section 7.2). Later we relax this requirement, delegating the measurement task only to nodes within the r-hop neighborhood of the current service host letting r modulate the accuracy *vs.* generated message overhead tradeoff.

Network-wide traffic measurements. Table 4 reports the performance of our practical implementation in terms of mean normalized cost and hopcount values when the current service host collects traffic measurements by every network node. Both practical instances turn out to be on average as accurate as their theoretical primitive; indeed, in most cases they provide solutions with cost within 2.5% of the optimal. This close match holds across different ISP network topologies and demand patterns suggesting that cDSMA is adequately robust to variations of network topology and traffic demand. More importantly, our practical schemes respond successfully to the different ways the underlying routing protocols transform the actual spatial demand distribution. As a result, a service user is expected to experience consistently close-to-optimal performance irrespective of the employed routing protocol. On the other hand, some increased confidence intervals suggest that a few nodes in the corresponding samples trap the migrating service and therefore increase the placement costs. Since the 1-median subgraphs of the experiments in Table 4 are carried out with at most 6% of the respective network nodes, a slight increase of the subgraph size is tolerable and could improve the quality of the solution.

$cDSMA_{SP}$ typically needs only slightly more hops to reach the final host-node than $cDSMA_{TH}$, even if the spatial stretch of the nodes it selects each time is upper bounded (see Section 7.2). This stands in agreement with fig. 6, where, even for higher asymmetry (s=2), the number of G_{Host} nodes under $cDSMA_{TH}$ that lies further than the inherent bound of $cDSMA_{SP}$ is negligible. Accordingly, the service migration hops under $cDSMA_{MP}$ are pretty much the same with those under $cDSMA_{TH}$. Overall and with respect to the network diameter and the $\lceil |G_{Host}| \rceil$ size, both cDSMA practical instances take no more than three hops, on average, to reach their final location. Effectively, this minimizes the installation costs on intermediate hosts along the migration path and the overhead of host advertisements/measurement reports.

Traffic measurements within the R_{msr}-hop neighborhood. We repeat the same experiments; service advertisement takes place like before but now the service host-node prompts only those nodes lying within R_{msr} hops to communicate their traffic measurement values. Table 5 captures the tradeoff between the achieved algorithm accuracy and generated overhead in terms of measurement reporting messages. The latter is quantified by the average number of the path-recording dedicated messages (*Dd msg*) that serve as the cDSMA input. The obvious goal is to bound them at the expense of a negligible performance penalty. We also report $\Delta(av.msg)$, which is the difference between the average of the *total* number of messages when the traffic measurements are performed by all nodes and by nodes within R_{msr} hops, respectively. Table 5 reveals that both practical instances of cDSMA preserve high performance standards even when traffic measurements are spatially restricted. Although we keep the α percentages constant and equal to the corresponding ones employed in Table 2 and 4 experiments, the G_{Host} size varies across iterations being upper-bounded by $\lceil |G_{Host}| \rceil$. This affects the accuracy of our implementation for small R_{msr} values. Nevertheless, R_{msr} values of four or five hops suffice to achieve performance that lies on average within 3.5% of the optimal for both ISP topologies and demand dynamics.

Although the R_{msr} bound, in principle, hurts the cDSMA advantage of longer migration hops towards the final location, the resulting penalty is no more than one hop, on average, when compared to the network-wide measurement case. In terms of message overhead this means that we need to bear at most one more service advertisement task. On the other hand, as R_{msr} scales, the hopcount to destination decreases and the number of measurement reporting nodes (*i.e.*, $|G_{Host} - \{Host\}|$) slightly increases. The average number of cDSMA dedicated messages remains practically unaffected and in the order of tens. The *total* number of messages under the bounded-measurement implementation decreases with R_{msr} in light of the fewer advertisements, whereas its counterpart under the network-wide measurements remains constant. This is clear in the $\Delta(av.msg)$ values, which suggest a substantial message number reduction for the former case, up to several hundreds for the recommended R_{msr} values of four or five.

9 Engineering properties

We conclude the cDSMA description outlining its capacity to operate in the presence of dynamic demand variations and/or network failures.

cDSMA and dynamic (spatial) demand: Since end-users may access the storage resources through both fixed and mobile devices, they may connect to different system nodes in the course of time. Hence, cDSMA will need to respond to *temporal* variations of the service demand. For a given service placement, there are two ways to track changes and accordingly trigger a new service migration process. The first option has the service host node perform regular (time-based) executions of the algorithm in "search

TABLE 4
Performance of practical implementation under the theoretical $\lceil |G_{Host}| \rceil$ values

	$cDSMA_{SP}$				$cDSMA_{MP}$			
	s=0		**s=1**		**s=0**		**s=1**	
Dataset id	$\beta(\lceil \lvert G_{Host} \rvert \rceil)$	h_m	$\beta(\lceil \lvert G_{Host} \rvert \rceil)$	h_m	$\beta(\lceil \lvert G_{Host} \rvert \rceil)$	h_m	$\beta(\lceil \lvert G_{Host} \rvert \rceil)$	h_m
36	1.0039±0.0152	1.50±0.36	1.0316±0.0145	1.80±0.31	1.0135±0.0219	1.13±0.31	1.0170±0.0131	1.37±0.06
35	1.0122±0.0122	1.30±0.40	1.0229±0.0210	1.30±0.17	1.0087±0.0111	1.10±0.22	1.0145±0.0123	1.41±.006
33	1.0378±0.0441	0.97±0.13	1.0461±0.0278	1.12±0.14	1.0244±0.0408	1.0±0.0	1.0185±0.0152	1.02±0.03
23	1.0132±0.0356	1.53±0.48	1.0255±0.0164	1.25±0.18	1.0±0.0	1.43±0.36	1.0123±0.0084	1.17±0.05
21	1.0391±0.0529	1.26±0.32	1.0339±0.0206	1.34±0.18	1.0±0.0	1.53±0.36	1.0122±0.0132	1.48±0.07
27	1.0±0.0	2.30±0.62	1.0016±0.0036	3.39±0.33	1.0±0.0	2.23±0.58	1.0018±0.0040	3.23±0.06
13	1.0165±0.0481	3.07±1.01	1.0160±0.0093	2.59±0.39	1.0±0.0	2.87±1.09	1.0105±0.0069	2.36±0.06
20	1.0144±0.0124	1.33±0.44	1.0311±0.0225	1.26±0.12	1.0279±0.0400	1.13±0.29	1.0055 ±0.0051	1.29±0.04
52	1.0091±0.0132	0.97±0.13	1.0103±0.0059	1.13±0.21	1.0045±0.0099	1.07±0.18	1.0076±0.0062	1.10±0.02
41	1.0154±0.0137	1.07±0.18	1.0153±0.0103	1.40±0.26	1.0151±0.0135	1.07±0.32	1.0092±0.0078	1.50±0.14
40	1.0119±0.0144	1.0±0.0	1.0194±0.0096	1.16±0.19	1.0149±0.0154	1.27±0.32	1.0127±0.0093	1.09±0.04
39	1.0144±0.0080	1.0±0.0	1.0195±0.0118	0.99±0.01	1.0125±0.0080	0.98±0.11	1.0096±0.0069	1.09±0.06

TABLE 5
Performance of practical implementation when R_{msr} bounds the number of measurement reporting nodes

$cDSMA_{SP}$

	Dataset 35				**Dataset 20**			
R_{msr}:	2	3	4	5	2	3	4	5
under uniform demand (s=0)								
$\beta_{R_{msr}}$	1.0275 ± 0.0541	1.0122±0.0122	1.0122±0.0122	1.0122±0.0122	1.0133±0.0126	1.0149±0.0124	1.0144±0.0124	1.0144±0.0124
h_m	1.65±0.53	1.45±0.43	1.30±0.40	1.30±0.40	2.20±0.61	1.65±0.50	1.55±0.48	1.35±0.44
$av(Dd\ msg)$	8 ± 1	8±1	8±1	8±1	11±1	12±1	13±1	13±1
$\Delta(av.msg)$	89 ± 1	108±1	122±1	122±1	278±1	567±1	618±1	723±1
under non-uniform demand (s=1)								
$\beta_{R_{msr}}$	1.0306±0.0217	1.0299±0.0210	1.0299±0.0210	1.0299±0.0210	1.0357±0.0228	1.0338±0.0225	1.0322±0.0228	1.0322±0.0229
h_m	1.83 ± 0.36	1.48±0.22	1.35±0.17	1.30±0.17	2.29±0.60	1.77±0.38	1.55±0.27	1.43±0.21
$av(Dd\ msg)$	7 ± 1	7±1	7±1	7±1	15±1	18±1	21±1	23±1
$\Delta(av.msg)$	65±1	99±1	111±1	116±1	115±1	385±1	498±1	559±1

$cDSMA_{MP}$

	Dataset 35				**Dataset 20**			
R_{msr}:	2	3	4	5	2	3	4	5
under uniform demand (s=0)								
$\beta_{R_{msr}}$	1.0429±0.0739	1.0087±0.0111	1.0087±0.0111	1.0087±0.0111	1.0420±0.0501	1.0340±0.0451	1.0314±0.0427	1.0349±0.0452
h_m	1.25±0.39	1.25±0.32	1.10±0.22	1.10±0.22	1.88±0.65	1.50±0.54	1.42±0.51	1.25±0.39
$av(Dd\ msg)$	6 ± 1	7±1	7±1	7±1	10±1	11±1	12±1	12±1
$\Delta(av.msg)$	89±1	88±1	103±1	103±1	195±1	393±1	434±1	523±1
under non-uniform demand (s=1)								
$\beta_{R_{msr}}$	1.2655±0.5016	1.0141±0.0123	1.0145±0.0123	1.0145±0.0123	1.0539±0.0138	1.0254±0.0170	1.0131±0.0099	1.0079±0.0066
h_m	2.02±0.19	1.75±0.10	1.56±0.08	1.51±0.08	2.21±0.18	1.91±0.12	1.65±0.08	1.56±0.07
$av(Dd\ msg)$	7 ± 1	8±1	8±1	8±1	15±1	21±1	23±1	26±1
$\Delta(av.msg)$	75±1	101±1	120±1	125±1	188±1	342±1	476±1	521±1

of" demand-shift incidents. The second one is the event-based execution of the algorithm, whereby reported demand changes from measurement-performing nodes trigger the execution of the cDSMA. In case the demand shift occurs while the migration process is in progress, cDSMA, thanks to its iterative operation, can identify *online* a new demand gradient generated by an emerging heavy hitter. In each iteration the global service demand is locally captured by the mapping mechanism and used as a new input.

cDSMA and fault tolerance: Standard fault tolerance mechanisms can be tailored for the cDSMA operation. To begin with, single replica schemes like cDSMA are more vulnerable to network failures than schemes supporting replication. cDSMA stays unaffected upon crashes of nodes that do not currently host the service. Otherwise, every IP-enabled node capable of hosting a service has to ensure the availability of one or more synchronized back-up nodes to serve as the Hot-Standby alternative hosts [28]. In each cDSMA iteration *i.e.*, intermediate or final service location, the current host can greedily nominate the second best solution within the 1-median subgraph as the back-up host that will be engaged if needed to restore the cDSMA normal operation. Most likely that node will lie in the physical proximity of the active host and thus, provide a low-cost alternative. On the other hand, cDSMA can cope with link failures provided that the network graph remains connected; the underlying routing protocol recovers the routes that involve broken links and the migration proceeds normally.

10 Related work

Out of the vast literature that studies placement problems under the lenses of discrete optimization we focus on the *distributed* solutions and identify two main approaches: those adopting a *facility location* approach [11] and those drawing on a *knapsack problem* formulation [13]. Placement problems can be cast in the knapsack framework when constrained storage is considered. More relevant to our service deployment scenario though is the former approach that attracts both theoretical and practical interest. The theoretical thread relates to the *approximability* of the facility location problem by distributed approaches. Algorithms are typically executed over a complete bipartite graph where

the m facilities and n client nodes communicate with each other in synchronous send-receive rounds. Moscibroda and Wattenhofer in [29] draw on a primal-dual approach earlier devised by Jain and Vazirani [30], to derive a distributed algorithm that trades-off the approximation ratio with the communication overhead assuming $O(logn)$ bits message size. More recently, Pandit and Pemmaraju [31] have derived an alternative distributed algorithm for the metric facility location that runs in k rounds achieving an $O(m^{2/\sqrt{k}} \cdot n^{3/\sqrt{k}})$-approximation.

Our work, on the other hand, comes under the broader family of heuristic algorithms. Though less mathematically rigorous, they are practically *implementable* and most often extremely effective. In most of the proposed solutions in this context, the available content/service facilities migrate towards their optimal location. In [32] the authors propose a joint optimization of content replication and placement, suited for wireless networks. Nevertheless, their methodology resembles the one we follow. They formulate a capacitated multi-commodity optimization problem and break it down to a multitude of single-commodity problems that seek to minimize the corresponding cost of one information item available at each facility. Each problem is solved mimicking a local search technique that relies on measurements of the demand queries each node serves. Accordingly, these measurements drive the content replication and hand-over to the facility's immediate neighbors.

Even closer to our work is the paper of Smaragdakis *et al.* [26] that proposes the R-ball heuristic. They reduce the original k-median problem in multiple smaller-scale 1-median problems solved within an area of *R*-hops from the current location of each service facility. This concept has been realized into a functional system by the authors of [33] who consider practical improvements to cope with churn events of p2p overlays. In Section 6.3 we have detailed how cDSMA compares with a similar, local-search oriented approach. A slightly different instance of the iterative service migration through locally-determined hops has been adopted by Oikonomou and Stavrakakis in [27]. They exploit the shortest-path tree structures induced on the network graph by the routing protocol operation to estimate upper bounds for the aggregate cost when the service migrates to its immediate neighbors. The process is therefore decelerated by those short migration hops.

11 Conclusion

In view of the proliferation of user-centric service instances across the Internet and the ever increasing in-network storage capabilities, we have developed a scalable and effective heuristic approach to deal with the complexity and limitations of their distributed placement. Our algorithm relies on node centrality insights to iteratively migrate service facilities towards near-optimal locations under various demand dynamics achieving very good accuracy and fast convergence. Most importantly, it lends to realistic protocol implementations that preserve its advantages regardless the employed routing policies and exhibit welcome scalability properties with respect to the number of the involved nodes and message overhead.

References

[1] T. Plagemann *et al.*, "From content distribution networks to content networks - issues and challenges," *Comput. Commun.*, vol. 29, no. 5, pp. 551–562, 2006.

[2] V. Jacobson *et al.*, "Networking named content," in *5th ACM CoNEXT 2009*, Rome, Italy, December 2009, pp. 1–12.

[3] A. Galis *et al.*, "Management architecture and systems for future internet networks," in *FIA Book : "Towards the Future Internet - A European Research Perspective"*, Prague, May 2009, pp. 112–122.

[4] E. Silva *et al.*, "Supporting dynamic service composition at runtime based on end-user requirements," in *Proc. 1st International Workshop on User-generated Services*, Stockholm, Sweden, 2009.

[5] [Online]. Available: https://developers.google.com/appengine/

[6] [Online]. Available: http://pipes.yahoo.com/pipes/

[7] J. C. Yelmo *et al.*, "A user-centric approach to service creation and delivery over next generation networks," *Com. Comm.*, vol. 34, 2011.

[8] V. Valancius *et al.*, "Greening the internet with nano data centers," in *Proc. 5th ACM CoNEXT*, Rome, Italy, 2009, pp. 37–48.

[9] D. Trossen, M. Sarela, and K. Sollins, "Arguments for an information-centric internetworking architecture," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 2, pp. 26–33, Apr. 2010.

[10] [Online]. Available: https://joindiaspora.com/

[11] P. Mirchandani and R.Francis, *Discrete location theory*, John Wiley and Sons, 1990.

[12] P. Pantazopoulos, M. Karaliopoulos, and I. Stavrakakis, "Centrality-driven scalable service migration," in *Proc. of the 23rd International Teletraffic Congress (ITC'11)*, San Francisco, California, USA, 2011.

[13] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York: Wiley, 1990.

[14] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1447 –1460, Dec. 2008.

[15] R. Solis-Oba, "Approximation algorithms for the k-median problem," ser. Lecture Notes in Computer Science, vol. 3484. Springer, 2006.

[16] M. E. J. Newman, "The Structure and Function of Complex Networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[17] P. Pantazopoulos, I. Stavrakakis, A. Passarella, and M. Conti, "Efficient social-aware content placement for opportunistic networks," in *IFIP/IEEE WONS*, Kranjska Gora, Slovenia, February, 3-5 2010.

[18] S. Jaiswal *et al.*, "Formal analysis of passive measurement inference techniques," in *IEEE INFOCOM'06*, Barcelona, Spain, 2006.

[19] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of Mathematical Sociology*, vol. 25, pp. 163–177, 2001.

[20] T. Kanungo *et al.*, "A local search approximation algorithm for k-means clustering," *Comput. Geom. Theory Appl.*, vol. 28, no. 2-3, pp. 89–112, Jun. 2004.

[21] A. L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct 1999.

[22] G. Siganos *et al.*, "Power laws and the as-level internet topology," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 514–524, 2003.

[23] J.-J. Pansiot, "mrinfo dataset." [Online]. Available: http://svnet.u-strasbg.fr/mrinfo/

[24] J.-J. Pansiot, P. Mérindol, B. Donnet, and O. Bonaventure, "Extracting intra-domain topology from mrinfo probing," in *Proc. Passive and Active Measurement Conference (PAM)*, April 2010.

[25] R. M. Karp and R. E. Tarjan, "Linear expected-time algorithms for connectivity problems (extended abstract)," in *ACM STOC '80*, Los Angeles, California, 1980, pp. 368–377.

[26] G. Smaragdakis, N. Laoutaris, K. Oikonomou, I. Stavrakakis, and A. Bestavros, "Distributed Server Migration for Scalable Internet Service Deployment," *IEEE/ACM Trans. Netw.*, (To Appear) 2012.

[27] K. Oikonomou and I. Stavrakakis, "Scalable service migration in autonomic network environments," *IEEE JSAC*, vol. 28, no. 1, pp. 84–94, 2010.

[28] T. Ise *et al.*, "Hot standby communications system," Patent US 4 700 348, 1987.

[29] T. Moscibroda and R. Wattenhofer, "Facility location: distributed approximation," in *PODC '05*, 2005, pp. 108–117.

[30] K. Jain and V. V. Vazirani, "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation," *Journal of ACM*, vol. 48, no. 2, 2001.

[31] S. Pandit and S. Pemmaraju, "Return of the primal-dual: distributed metric facilitylocation," in *ACM PODC '09*, 2009, pp. 180–189.

[32] C.-A. La *et al.*, "Content replication and placement in mobile networks," *IEEE JSAC Cooperative Networking (part II)*, [To Appear].

[33] T. Sproull and R. Chamberlain, "Distributed algorithms for the placement of network services." in *International Conference on Internet Computing*, Las Vegas Nevada, USA, July 2010.